(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 107609004 B
(45)授权公告日 2020.08.18

(21)申请号 201710601957.5

(22)申请日 2017.07.21

(65)同一申请的已公布的文献号
申请公布号 CN 107609004 A

(43)申请公布日 2018.01.19

(73)专利权人 深圳市小牛在线互联网信息咨询有限公司
地址 518000 广东省深圳市南山区南山街道高新园高新南一道富诚科技大厦四楼

(72)发明人 文忠湖

(74)专利代理机构 广州华进联合专利商标代理有限公司 44224
代理人 谢曲曲

(51)Int.Cl.
G06F 16/2458(2019.01)
G06F 16/2455(2019.01)
G06F 9/448(2018.01)
G06N 20/00(2019.01)

(56)对比文件
CN 106598868 A,2017.04.26,
CN 106534212 A,2017.03.22,
CN 104572043 A,2015.04.29,
US 2013007769 A1,2013.01.03,
何维.网站数据采集系统设计与实现.《中国优秀硕士学位论文全文数据库信息科技辑》.2017,全文.

审查员 熊菡

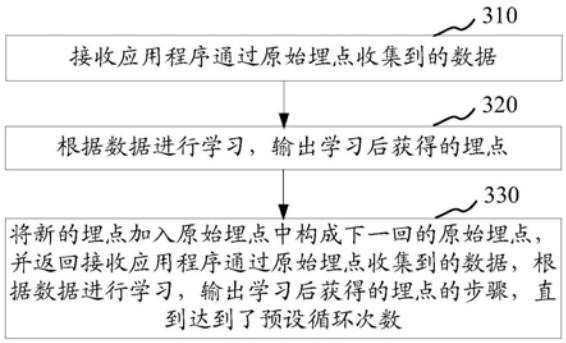
权利要求书2页 说明书10页 附图5页

(54)发明名称

应用程序埋点方法和装置、计算机设备和存储介质

(57)摘要

本发明涉及一种应用程序埋点方法和装置，服务器对根据原始埋点收集到的数据进行学习，输出新的埋点。将新的埋点加入原始埋点中构成下一回的原始埋点，并开始循环计算新的埋点。还提供了一种应用程序埋点方法，包括：安装了应用程序的终端接收服务器发送的下一回的原始埋点，下一回的原始埋点包括学习后获得的埋点，将下一回的原始埋点通过动态寻址程序插入到应用程序的类文件中，并打包运行更新后的应用程序。如此循环下去，从数据中不断学习新的埋点、对埋点进行自动更新，直到达到预设循环次数或者认为触发停止，全程不需要人工干预添加埋点，提高了对应用程序进行埋点的正确性，且能够更好的适应未来各复杂业务需求的迭代更新。



CN 107609004 B

1.一种应用程序埋点方法,其特征在于,所述方法包括:

接收应用程序通过原始埋点收集到的数据;

根据所述数据进行学习,输出学习后获得的埋点;所述根据所述数据进行学习,输出学习后获得的埋点,包括:根据所述数据采用动态学习模型,计算用户自身的用户行为与所述原始埋点之间的关联度,计算所述用户行为发生的概率;根据所述关联度和所述概率从所述原始埋点中剔除无效埋点并筛选出新的埋点;将剔除了无效埋点并加入了新的埋点的原始埋点输出作为所述学习后获得的埋点;

将所述学习后获得的埋点作为下一回的原始埋点,并返回所述接收应用程序通过原始埋点收集到的数据,根据所述数据进行学习,输出学习后获得的埋点的步骤,直到达到了预设循环次数。

2.根据权利要求1所述的方法,其特征在于,在所述接收应用程序通过原始埋点收集到的数据之后,还包括:

根据预设条件通过数据匹配从所述数据中剔除不符合所述预设条件的数据;

对剔除了不符合的数据之后获得的数据进行数据分析和数据挖掘;

输出数据分析和数据挖掘的结果。

3.根据权利要求1所述的方法,其特征在于,在所述根据所述数据进行学习,输出学习后获得的埋点之后,还包括:

根据业务需求在所述学习后获得的埋点中添加自定义埋点,构成下一回的原始埋点。

4.一种应用程序埋点方法,其特征在于,包括:

接收服务器发送的下一回的原始埋点,所述下一回的原始埋点包括学习后获得的埋点;所述学习后获得的埋点为对应用程序通过原始埋点收集到的数据采用动态学习模型,计算用户自身的用户行为与所述原始埋点之间的关联度,计算所述用户行为发生的概率;根据所述关联度和所述概率从所述原始埋点中剔除无效埋点并筛选出新的埋点所得;

将所述下一回的原始埋点通过动态寻址程序插入到应用程序的类文件中,并打包运行更新后的应用程序。

5.根据权利要求4所述的方法,其特征在于,所述将所述下一回的原始埋点通过动态寻址程序插入到应用程序的类文件中,并打包运行更新后的应用程序,包括:

将所述下一回的原始埋点进行结构解析,并调用动态寻址脚本;

通过所述动态寻址脚本对所述原始埋点解析出的代码结构进行代码循环匹配字节码;

通过命令重新打包构建新的应用程序,并运行更新后的应用程序。

6.一种应用程序埋点装置,其特征在于,包括:

数据接收模块,用于接收应用程序通过原始埋点收集到的数据;

学习模块,用于根据所述数据进行学习,输出学习后获得的埋点;还用于根据所述数据采用动态学习模型,计算用户自身的用户行为与所述原始埋点之间的关联度,计算所述用户行为发生的概率;根据所述关联度和所述概率从所述原始埋点中剔除无效埋点并筛选出新的埋点;将剔除了无效埋点并加入了新的埋点的原始埋点输出作为所述学习后获得的埋点;

循环模块,用于将所述学习后获得的埋点作为下一回的原始埋点,并返回所述接收应用程序通过原始埋点收集到的数据,根据所述数据进行学习,输出学习后获得的埋点的步

骤，直到达到了预设循环次数。

7.根据权利要求6所述的装置，其特征在于，还包括：

数据剔除模块，用于根据预设条件通过数据匹配从数据中剔除不符合预设条件的数据；

数据分析和数据挖掘模块，用于对剔除了不符合的数据之后获得的数据进行数据分析和数据挖掘；

输出模块，用于输出数据分析和数据挖掘的结果。

8.根据权利要求6所述的装置，其特征在于，还包括：

自定义埋点添加模块，用于根据业务需求在学习后获得的埋点中添加自定义埋点，构成下一回的原始埋点。

9.一种计算机可读存储介质，其上存储有计算机程序，其特征在于，该程序被处理器执行时实现如权利要求1至5中任一项所述的应用程序埋点方法。

10.一种计算机设备，所述计算机设备包括存储器，处理器及存储在所述存储器上并可在所述处理器上运行的计算机程序，其特征在于，所述处理器执行所述计算机程序时实现如权利要求1至5中任一项所述的应用程序埋点方法。

应用程序埋点方法和装置、计算机设备和存储介质

技术领域

[0001] 本发明涉及计算机技术领域，特别是涉及一种应用程序埋点方法和装置、计算机设备和存储介质。

背景技术

[0002] 目前埋点主要用于计算机技术领域，它可以通过对指定程序的数据埋点，采集数据、分析数据，为用户提供高质量、智能性的便民服务。

[0003] 传统技术中，应用程序埋点方法是随着业务产品的迭代更新，收集点也跟随业务的变动而进行动态迭代更新，代码层上人工修改添加数据收集点容易出现收集点丢失、冗余不全等问题，导致数据收集统计分析不准确，误导企业分析以及用户体验等问题。例如，技术人员在埋点的时候，必须先清楚要收集的是什么数据，这些数据用来做什么，前期准备时间过长，埋点过程较慢，且极易出错。其次，在数据分析过程中，如果发现数据有问题，纠正则需要重新进行埋点，使得之前的工作变成无用功。随着企业公司日益快速的成长，其业务复杂繁多且频繁迭代，导致在人工数据收集工作和数据分析工作中，极易出现收集点丢失、收集数据不准确的问题。

发明内容

[0004] 基于此，有必要针对上述技术问题，提供一种能提高埋点准确性的应用程序埋点方法和装置、计算机设备和存储介质。

[0005] 一种应用程序埋点方法，所述方法包括：

[0006] 接收应用程序通过原始埋点收集到的数据；

[0007] 根据所述数据进行学习，输出新的埋点；

[0008] 将所述新的埋点加入所述原始埋点中构成下一回的原始埋点，并返回所述接收应用程序通过原始埋点收集到的数据，根据所述数据进行学习，输出新的埋点的步骤，直到达到了预设循环次数。

[0009] 在其中一个实施例中，所述根据所述数据进行学习，输出新的埋点，包括：

[0010] 根据所述数据采用动态学习模型，计算用户自身的用户行为与所述原始埋点之间的关联度，计算所述用户行为发生的概率；

[0011] 根据所述关联度和所述概率从所述原始埋点中剔除无效埋点并筛选出新的埋点；

[0012] 将所述新的埋点输出。

[0013] 在其中一个实施例中，在所述接收应用程序通过原始埋点收集到的数据之后，还包括：

[0014] 根据预设条件通过数据匹配从所述数据中剔除不符合所述预设条件的数据；

[0015] 对剔除了不符合的数据之后获得的数据进行数据分析和数据挖掘；

[0016] 输出数据分析和数据挖掘的结果。

[0017] 在其中一个实施例中，在所述根据所述数据进行学习，输出新的埋点之后，还包

括：

[0018] 根据业务需求在所述新的埋点中添加自定义埋点，构成下一回的原始埋点。

[0019] 一种应用程序埋点方法，包括：

[0020] 接收服务器发送的下一回的原始埋点，所述下一回的原始埋点包括上一回的原始埋点和根据所述原始埋点进行学习输出的新的埋点；

[0021] 将所述下一回的原始埋点通过动态寻址程序插入到应用程序的类文件中，并打包运行更新后的应用程序。

[0022] 在其中一个实施例中，所述将所述目标埋点通过动态寻址程序插入到应用程序的类文件中，并打包运行更新后的应用程序，包括：

[0023] 将所述下一回的原始埋点进行结构解析，并调用动态寻址脚本；

[0024] 通过所述动态寻址脚本对所述原始埋点解析出的代码结构进行代码循环匹配字节码；

[0025] 通过命令重新打包构建新的应用程序，并运行更新后的应用程序。

[0026] 一种应用程序埋点装置，包括：

[0027] 数据接收模块，用于接收应用程序通过原始埋点收集到的数据；

[0028] 学习模块，用于根据所述数据进行学习，输出新的埋点；

[0029] 循环模块，用于将所述新的埋点加入所述原始埋点中构成下一回的原始埋点，并返回所述接收应用程序通过原始埋点收集到的数据，根据所述数据进行学习，输出新的埋点的步骤，直到达到了预设循环次数。

[0030] 在其中一个实施例中，学习模块还用于：根据所述数据采用动态学习模型，计算用户自身的用户行为与所述原始埋点之间的关联度，计算所述用户行为发生的概率；根据所述关联度和所述概率从所述原始埋点中剔除无效埋点并筛选出新的埋点；将所述新的埋点输出。

[0031] 一种计算机可读存储介质，其上存储有计算机程序，该程序被处理器执行时实现以下步骤：

[0032] 接收应用程序通过原始埋点收集到的数据；

[0033] 根据所述数据进行学习，输出新的埋点；

[0034] 将所述新的埋点加入所述原始埋点中构成下一回的原始埋点，并返回所述接收应用程序通过原始埋点收集到的数据，根据所述数据进行学习，输出新的埋点的步骤，直到达到了预设循环次数。

[0035] 一种计算机设备，所述计算机设备包括存储器，处理器及存储在所述存储器上并可在所述处理器上运行的计算机程序，所述处理器执行所述计算机程序时实现以下步骤：

[0036] 接收应用程序通过原始埋点收集到的数据；

[0037] 根据所述数据进行学习，输出新的埋点；

[0038] 将所述新的埋点加入所述原始埋点中构成下一回的原始埋点，并返回所述接收应用程序通过原始埋点收集到的数据，根据所述数据进行学习，输出新的埋点的步骤，直到达到了预设循环次数。

[0039] 上述应用程序埋点方法和装置、计算机设备和存储介质，对根据原始埋点收集到的数据进行学习，输出新的埋点。将新的埋点加入原始埋点中构成下一回的原始埋点，并开

始循环计算新的埋点。从数据中不断学习新的埋点，这样能够应付未来各复杂业务需求迭代更新。不需要人工参与，就可以自动化、智能化地获取到精确性较高的埋点。进而提高对应用程序进行埋点的正确性。

附图说明

[0040] 图1为一个实施例中应用程序埋点方法的应用场景示意图；

[0041] 图2为一个实施例中服务器的内部结构图；

[0042] 图3一个实施例中应用程序埋点方法的流程图；

[0043] 图4为图3中根据数据进行学习的流程图；

[0044] 图5为对收集到的数据进行数据分析和数据挖掘的流程图；

[0045] 图6为一个实施例中应用程序埋点方法的流程图；

[0046] 图7为一个实施例中应用程序埋点装置的结构示意图；

[0047] 图8为一个实施例中应用程序埋点装置的结构示意图；

[0048] 图9为一个实施例中应用程序埋点装置的结构示意图；

[0049] 图10为一个实施例中应用程序埋点装置的结构示意图。

具体实施方式

[0050] 为了使本发明的目的、技术方案及优点更加清楚明白，以下结合附图及实施例，对本发明进行进一步详细说明。应当理解，此处所描述的具体实施例仅仅用以解释本发明，并不用于限定本发明。

[0051] 可以理解，本发明所使用的术语“第一”、“第二”等可在本文中用于描述各种元件，但这些元件不受这些术语限制。这些术语仅用于将第一个元件与另一个元件区分。举例来说，在不脱离本发明的范围的情况下，可以将第一客户端称为第二客户端，且类似地，可将第二客户端称为第一客户端。第一客户端和第二客户端两者都是客户端，但其不是同一客户端。

[0052] 如图1所示，本申请实施例所提供的应用程序埋点方法可应用如图1所示的环境中，包括服务器102和终端104。其中，服务器102与终端104可通过无线通信方式进行数据通信。服务器102上可安装Windows操作系统、Linux操作系统等。终端104可为个人计算机、智能手机、平板电脑、个人数字助理、穿戴式设备等。无线通信方式可为WIFI、以太网等方式。终端102上的应用程序通过原始埋点收集数据，将收集到的数据上传至服务器102。服务器102接收终端104上传的数据，对数据进行学习输出新的埋点。将新的埋点加入原始埋点中构成下一回的原始埋点，并返回接收应用程序通过原始埋点收集到的数据，根据数据进行学习，输出新的埋点的步骤，直到达到了预设循环次数。

[0053] 如图2所示，在一个实施例中，还提供了一种服务器，该服务器包括通过系统总线连接的处理器、非易失性存储介质、内存储器、网络接口，非易失性存储介质中存储有操作系统和一种计算机程序，该计算机程序被处理器执行时可用于执行一种应用程序埋点方法。该处理器用于提高计算和控制能力，支撑整个服务器的运行。内存存储器用于为非易失性存储介质中的计算机程序提供运行环境，该内存存储器中可存储有计算机可读指令，该计算机可读指令被处理器执行时，可使得该处理器执行一种应用程序埋点方法。网络接口

用于与终端进行网络通信，接收或者发送数据，例如接收终端发送的数据，以及向终端发送埋点等。

[0054] 如图3所示，在一个实施例中，提供了一种应用程序埋点方法，该方法以应用于如图1所示的服务器中进行举例说明，包括：

[0055] 步骤310，接收应用程序通过原始埋点收集到的数据。

[0056] 数据埋点是产品数据分析的基础，一般用于对用户行为的监控和分析。目前常见的前端埋点技术有三类：在某个控件操作发生时通过预先写好的代码来进行数据的代码埋点；通过可视化界面配置控件操作与事件发生关系的可视化埋点；先收集所有数据再在后端筛选需要分析的对象的“无埋点”。

[0057] 埋点的时机主要分两个时机，一个是静态模式下，就是每次重新部署客户端软件上传到各个渠道时，把原始埋点在编译客户端代码时埋入客户端软件。另外一个是动态模式下，客户端软件运行时，主动触发客户端软件更新埋点。这两种埋点时机可以同时并行。

[0058] 在静态模式下部署了埋点后的应用程序在程序运行时，通过原始埋点收集原始埋点对应的数据。应用程序再将收集到的原始埋点上传至服务器中。具体的，埋点的类型一般有两种，非业务类型和业务类型。非业务类型埋点也可以称为固定类型埋点，例如，在网络协议传输过程中埋点、在生命周期的回调函数进行埋点、对用户点击按钮或者滑动进行埋点等。业务类型埋点，例如可以是对用户注册行为的埋点、对用户忘记密码行为的埋点、对用户下单行为的埋点、对用户支付充值行为的埋点等。

[0059] 埋点后的应用程序在程序运行时，通过原始埋点收集原始埋点对应的数据。例如，原始埋点可以是用户的支付行为，则当应用程序运行时，收集与用户的支付行为相关的数据。该数据即为应用程序通过原始埋点收集到的数据。具体的，收集到的数据可以是“大学生小明通过微信进行支付，买了一台华为手机”。

[0060] 步骤320，根据数据进行学习，输出学习后获得的埋点。

[0061] 服务器根据从应用程序接收的数据进行学习。具体的，采用贝叶斯网络学习模型进行学习，当然，也可以采用其他学习模型来进行学习。将所有的埋点当成一个一个的节点，每个节点之间存在着有方向的依赖关系。把这些节点之间的有方向的依赖关系作为一个向量，那么这些向量最终会构成一个向量图，向量有方向和长度，每一条向量展现出每一个节点的有方向的依赖关系。根据这些依赖关系，预测学习后获得的埋点。例如，将埋点A和埋点B当成节点，这两个节点均与节点C之间有连线，这说明节点C与节点A和节点B之间存在一定的关系。若节点C与节点D之间的向量指向节点D，那么节点D与节点A和节点B之间也存在一定的关系。根据节点C与节点A和节点B之间的关系，来预测节点C是否为新的埋点。根据节点D与节点A和节点B之间的关系，来预测节点D是否为学习后获得的埋点。

[0062] 步骤330，将新的埋点加入原始埋点中构成下一回的原始埋点，并返回接收应用程序通过原始埋点收集到的数据，根据数据进行学习，输出学习后获得的埋点的步骤，直到达到了预设循环次数。

[0063] 服务器将经过上述学习模型进行学习后，获取预测到的学习后获得的埋点，将学习后获得的埋点加入第一次的原始埋点中，构成下一回的原始埋点。并将这些下一回的原始埋点插入到应用程序中，当应用程序开始运行时，则根据新插入的埋点进行采集数据。如此循环，直至达到了预设循环次数，或者进行手动终止。

[0064] 在本实施例中，对根据原始埋点收集到的数据进行学习，输出新的埋点。将新的埋点加入原始埋点中构成下一回的原始埋点，并开始循环计算新的埋点。从数据中不断学习新的埋点，这样能够应付未来各复杂业务需求迭代更新。不需要人工参与，就可以自动化、智能化地获取到精确性较高的埋点。进而提高对应用程序进行埋点的正确性。

[0065] 在一个实施例中，如图4所示，根据数据进行学习，输出学习后获得的埋点，包括：

[0066] 步骤322，根据数据采用动态学习模型，计算用户自身的用户行为与原始埋点之间的关联度，计算用户行为发生的概率。

[0067] 应用程序将根据原始埋点所收集到的与原始埋点相关的数据进行上传至服务器中，服务器接收这些数据进行学习。具体的，可以采用贝叶斯网络学习模型进行学习，当然也可以采用其他学习模型进行学习。具体的，贝叶斯网络学习模型的原理为：当不能准确知悉一个事物的本质时，可以依靠与事物特定本质相关的事件出现的多少去判断其本质属性的概率。用数学语言表达就是：支持某项属性的事件发生得愈多，则该属性成立的可能性就愈大。

[0068] 例如，将所有的埋点当成一个一个的节点，这些节点之间存在一定的关系。用贝叶斯网络学习模型计算用户自身的用户行为与原始埋点之间的关联度，计算用户行为发生的概率。本来有一个埋点A指的是用户的“支付”行为，从应用程序通过这个埋点A获取到的埋点数据中，去按照业务规则去计算新的埋点的概率。按照业务规则从支付这个埋点去定义“支付结果”与“支付”匹配的概率是100％，且“支付结果”与“支付”两个埋点之间的向量长度为1。定义“绑卡”与“支付”匹配的概率为80％，且“绑卡”与“支付”两个埋点之间的向量长度为2。定义“用户登录”与“支付”的概率为60％，且“用户登录”与“支付”两个埋点之间的向量长度为3。向量长度表示两者之间的关联度，向量长度越小表示关联度很高。当然，根据不同公司的不同业务规则定义的概率、向量长度及真实会发生的概率都会不同。

[0069] 假设之前的原始埋点中有“支付”也有“退款”，而经过对通过原始埋点收集到的数据中进行学习，计算出“退款”与“支付”匹配的概率只为20％。不符合规定，因为规定了概率超过60％的才是真实会发生的用户行为。因此，需要将这个无效埋点—“退款”从原始埋点中剔除。

[0070] 步骤324，根据关联度和概率从原始埋点中剔除无效埋点并筛选出新的埋点。

[0071] 假设规定了概率超过60％的才是真实会发生的，那么上述“支付结果”、“绑卡”、“用户登录”这三个用户行为都是会真实发生的，可以作为新的埋点。计算出“退款”与“支付”匹配的概率只为20％。不符合规定，因此需要将这个效埋点—“退款”从原始埋点中剔除。从而实现了根据关联度和概率从原始埋点中剔除无效埋点并筛选出新的埋点。

[0072] 步骤326，将剔除了无效埋点并加入了新的埋点的原始埋点的埋点输出。

[0073] 将服务器通过贝叶斯网络学习模型学习出的新的埋点，加入剔除了无效埋点之后的埋点中，将最终更新之后的埋点输出至应用程序所在的终端。

[0074] 在本实施例中，服务器对接收这些数据采用学习模型进行学习，计算出不同埋点和预测的埋点的概率和关联度，根据公司不同的业务需求定义筛选或提出埋点的概率和关联度条件，根据该条件去对埋点和预测的埋点进行筛选或剔除。从而实现了根据关联度和概率从原始埋点中剔除无效埋点并筛选出新的埋点。最后，将剔除了无效埋点并加入了新的埋点的原始埋点的埋点输出。服务器对每一回接收这些数据都采用学习模型进行学习，

这样就可以不断地将原始埋点进行更新，剔除了无效埋点并加入了新的埋点。保证了埋点的时效性和准确性，可以及时根据所收集到的数据的变化来对埋点的方向进行调整，及时感知市场的变化，保证收集到的数据的准确性，进而保证后续对数据进行数据分析和数据挖掘的结果的准确性。

[0075] 在一个实施例中，如图5所示，在接收应用程序通过原始埋点收集到的数据之后，还包括：

[0076] 步骤340，根据预设条件通过数据匹配从数据中剔除不符合预设条件的数据。

[0077] 埋点后的应用程序在程序运行时，通过原始埋点收集原始埋点对应的数据。例如，原始埋点可以是用户的支付行为，则当应用程序运行时，收集与用户的支付行为相关的数据。该数据即为应用程序通过原始埋点收集到的数据。具体的，收集到的数据可以是“大学生小明通过微信进行支付，买了一台华为手机”。通过原始埋点收集原始埋点对应的数据范围太广，因此在实际中公司会根据业务需求定义预设条件，去从收集到的数据中剔除不符合预设条件的数据，留下符合预设条件的数据。

[0078] 具体的，是通过数据匹配来进行剔除。在对收集到的数据进行数据匹配时，需要预先定义数据匹配的规则，规则可以为不止一条，根据这些规则剔除不符合的数据。例如，公司需要获取本周的支付进出次数的话，如果公司设置的条件一个是支付行为发生的时间是在每天下班到凌晨即(pm6:00-pm12:00)，另一个是这个支付行为对应的用户需是注册用户，即这个支付行为不能是在游客模式下进行的。那么通过这两个条件，既可以实现数据匹配，从而剔除的不符合预设条件的数据，留下符合预设条件的数据。

[0079] 步骤350，对剔除了不符合的数据之后获得的数据进行数据分析和数据挖掘。

[0080] 步骤360，输出数据分析和数据挖掘的结果。

[0081] 对剔除了不符合的数据之后获得的数据进行数据分析和数据挖掘，将结果输出。例如，根据公司设置的条件对通过原始埋点获取的数据进行剔除，留下满则预设条件的数据。留下的数据都满足支付行为发生的时间是在每天下班到凌晨即(pm6:00-pm12:00)，且这个支付行为对应的用户是注册用户这两个条件。然后对这些留下的数据进行分析就可以获取公司定义的本周的支付进出次数的结果。

[0082] 在本实施例中，因为根据埋点所收集到的数据量很大，服务器便对通过原始埋点所收集到的数据先进行筛选，筛选的条件可以根据公司的业务需求或者公司想要分析的方向去人为设定。如此，便可以实现个性化自定义。对剔除了不符合条件的数据进行分析，这样大大提高了分析结果的准确性，且减少了服务器的工作量，提高了效率。

[0083] 在一个实施例中，在根据数据进行学习，输出学习后获得的埋点之后，还包括：根据业务需求在学习后获得的埋点中添加自定义埋点，构成下一回的原始埋点。

[0084] 在本实施例中，在上述通过贝叶斯网络学习模型输出学习后获得的埋点之后，可能因为公司业务发展的需要，需要采集一些与之前的埋点的领域不同的数据。因此，就需要在下一次通过原始埋点进行数据采集时候，加入自定义埋点。例如，之前的原始埋点是关于支付、实物买卖等领域，而现在根据公司业务的发展，需要开展运费险业务，因此就需要加入一些与运费险相关的埋点来进行数据采集。这样可以及时调整埋点的方向。

[0085] 在一个实施例中，如图6所示，还提供了一种应用程序埋点方法，该方法以应用于如图1所示的终端中进行举例说明，包括：

[0086] 步骤610,接收服务器发送的下一回的原始埋点,下一回的原始埋点包括学习后获得的埋点。

[0087] 应用程序在开发阶段就已经预先通过代码插入了原始埋点,这样当终端上安装了应用程序后,在应用程序运行时候,应用程序即开始通过原始埋点进行数据采集。应用程序收集了数据后,例如可以定期上传至服务器端进行保存,也可以当收集到的数据达到一定的数据量再上传只服务器进行保存,或者人工主动通过服务器来获取数据。服务器对收集的数据进行学习后,输出了学习后获得的埋点,将这些学习后获得的埋点发送至应用程序所在的终端,终端接收服务器发送的埋点,作为下一回的原始埋点开始新一轮的收集数据。

[0088] 步骤620,将下一回的原始埋点通过动态寻址程序插入到应用程序的类文件中,并打包运行更新后的应用程序。

[0089] 终端接收了服务器发送的埋点,作为下一回收集数据的原始埋点。终端将对这些下一回的原始埋点通过动态寻址程序插入到应用程序的类文件中。具体的,将下一回的原始埋点进行结构解析并调用动态寻址脚本,再通过动态寻址脚本对原始埋点解析出的代码结构进行代码循环匹配字节码。匹配成功则通过命令重新打包构建新的应用程序,并运行更新后的应用程序。

[0090] 在本实施例中,安装了应用程序的终端在服务器学习出埋点之后,就将这些学习之后获得的埋点通过动态编译技术插入至应用程序中,将应用程序更新。应用程序将根据这些学习之后的埋点开始新一轮的收集数据。如此循环下去,不断地对埋点进行自动更新,直到达到预设循环次数或者认为触发停止,不需要人工干预添加埋点。使用动态编译技术可以实现直接将更新后的埋点插入至程序中,对程序进行更新后运行更新后的程序。传统的方法,在埋点之前,技术人员在埋点的时候,必须先清楚要收集的是什么数据,这些数据用来做什么,前期准备时间过长,埋点过程较慢,且极易出错。其次,在数据分析过程中,如果发现数据有问题,纠正则需要人工重新进行埋点,使得之前的工作变成无用功。

[0091] 在一个实施例中,将下一回的原始埋点通过动态寻址程序插入到应用程序的类文件中,并打包运行更新后的应用程序,包括:将下一回的原始埋点进行结构解析,并调用动态寻址脚本;通过动态寻址脚本对原始埋点解析出的代码结构进行代码循环匹配字节码;通过命令重新打包构建新的应用程序,并运行更新后的应用程序。

[0092] 在本实施例中,具体的,读取埋点对应的数据列表文件进行结构解析,解析成可让编译脚本识别的代码结构作为前置数据。引入动态寻址技术的框架脚本,可选择CGLIB、javassist或者ASM。CGLIB(Code Generation Library)是一个功能强大,高性能的代码生成包。它为没有实现接口的类提供代理,为JDK的动态代理提供了很好的补充。Javassist是一个开源的分析、编辑和创建Java字节码的类库。ASM(Assembly Language,汇编语言)是一个Java字节码操控框架。它能够以二进制形式修改已有类或者动态生成类。ASM可以直接产生二进制class文件,也可以在类被加载入Java虚拟机之前动态改变类行为。ASM从类文件中读入信息后,能够改变类行为,分析类信息,甚至能够根据用户要求生成新类。

[0093] 通过动态寻址脚本根据列表文件解析出来的代码结构进行循环匹配class字节码,匹配成功则通过寻址技术切入到class字节码,反之则触发错误存放机制。class字节码文件是根据JVM(Java Virtual Machine,Java虚拟机)规范中规定的字节码组织规则生成的。全局寻址完毕则进行目录构建后动态载入动态埋点的框架。

[0094] 通过Dex命令进行重新打包构建新的应用程序，再根据密文进行签名产生可运行的应用程序，最后运行更新后的应用程序。

[0095] 在一个实施例中，如图7所示，还提供了一种应用程序埋点装置700，包括：数据接收模块710、学习模块720及循环模块730。

[0096] 数据接收模块710，用于接收应用程序通过原始埋点收集到的数据。

[0097] 学习模块720，用于根据数据进行学习，输出学习后获得的埋点。

[0098] 循环模块730，用于将学习后获得的埋点作为下一回的原始埋点，并返回接收应用程序通过原始埋点收集到的数据，根据数据进行学习，输出学习后获得的埋点的步骤，直到达到了预设循环次数。

[0099] 在一个实施例中，学习模块720还用于：根据数据采用动态学习模型，计算用户自身的用户行为与原始埋点之间的关联度，计算用户行为发生的概率；根据关联度和概率从原始埋点中剔除无效埋点并筛选出新的埋点；将剔除了无效埋点并加入了新的埋点的原始埋点输出。

[0100] 在一个实施例中，如图8所示，一种应用程序埋点装置700，还包括：数据剔除模块740、数据分析和数据挖掘模块750及输出模块760。

[0101] 数据剔除模块740，用于根据预设条件通过数据匹配从数据中剔除不符合预设条件的数据；

[0102] 数据分析和数据挖掘模块750，用于对剔除了不符合的数据之后获得的数据进行数据分析和数据挖掘；

[0103] 输出模块760，用于输出数据分析和数据挖掘的结果。

[0104] 在一个实施例中，如图9所示，一种应用程序埋点装置700，还包括：自定义埋点添加模块770，用于根据业务需求在学习后获得的埋点中添加自定义埋点，构成下一回的原始埋点。

[0105] 在一个实施例中，如图10所示，一种应用程序埋点装置800，包括：原始埋点接受模块810和原始埋点插入模块820。

[0106] 原始埋点接收模块810，用于接收服务器发送的下一回的原始埋点，下一回的原始埋点包括学习后获得的埋点；

[0107] 原始埋点插入模块820，用于将下一回的原始埋点通过动态寻址程序插入到应用程序的类文件中，并打包运行更新后的应用程序。

[0108] 在一个实施例中，原始埋点插入模块820还用于：将下一回的原始埋点进行结构解析，并调用动态寻址脚本；通过动态寻址脚本对原始埋点解析出的代码结构进行代码循环匹配字节码；通过命令重新打包构建新的应用程序，并运行更新后的应用程序。

[0109] 在一个实施例中，还提供了一种计算机可读存储介质，其上存储有计算机程序，该程序被处理器执行时实现以下步骤：接收应用程序通过原始埋点收集到的数据；根据数据进行学习，输出学习后获得的埋点；将学习后获得的埋点加入原始埋点中构成下一回的原始埋点，并返回接收应用程序通过原始埋点收集到的数据，根据数据进行学习，输出学习后获得的埋点的步骤，直到达到了预设循环次数。

[0110] 在一个实施例中，上述程序被处理器执行时还实现以下步骤：根据数据进行学习，输出学习后获得的埋点，包括：根据数据采用动态学习模型，计算用户自身的用户行为与原

始埋点之间的关联度，计算用户行为发生的概率；根据关联度和概率从原始埋点中剔除无效埋点并筛选出新的埋点；将剔除了无效埋点并加入了新的埋点的原始埋点输出。

[0111] 在一个实施例中，上述程序被处理器执行时还实现以下步骤：在接收应用程序通过原始埋点收集到的数据之后，还包括：根据预设条件通过数据匹配从数据中剔除不符合预设条件的数据；对剔除了不符合的数据之后获得的数据进行数据分析和数据挖掘；输出数据分析和数据挖掘的结果。

[0112] 在一个实施例中，上述程序被处理器执行时还实现以下步骤：在根据数据进行学习，输出学习后获得的埋点之后，还包括：根据业务需求在学习后获得的埋点中添加自定义埋点，构成下一回的原始埋点。

[0113] 在一个实施例中，上述程序被处理器执行时还实现以下步骤：一种应用程序埋点方法，包括：接收服务器发送的下一回的原始埋点，下一回的原始埋点包括学习后获得的埋点；将下一回的原始埋点通过动态寻址程序插入到应用程序的类文件中，并打包运行更新后的应用程序。

[0114] 在一个实施例中，上述程序被处理器执行时还实现以下步骤：将下一回的原始埋点通过动态寻址程序插入到应用程序的类文件中，并打包运行更新后的应用程序，包括：将下一回的原始埋点进行结构解析，并调用动态寻址脚本；通过动态寻址脚本对原始埋点解析出的代码结构进行代码循环匹配字节码；通过命令重新打包构建新的应用程序，并运行更新后的应用程序。

[0115] 在一个实施例中，还提供了一种计算机设备，该计算机设备包括存储器，处理器及存储在存储器上并可在处理器上运行的计算机程序，处理器执行计算机程序时实现以下步骤：接收应用程序通过原始埋点收集到的数据；根据数据进行学习，输出学习后获得的埋点；将学习后获得的埋点作为下一回的原始埋点，并返回接收应用程序通过原始埋点收集到的数据，根据数据进行学习，输出学习后获得的埋点的步骤，直到达到了预设循环次数。

[0116] 在一个实施例中，上述处理器执行计算机程序时还实现以下步骤：根据数据进行学习，输出学习后获得的埋点，包括：根据数据采用动态学习模型，计算用户自身的用户行为与原始埋点之间的关联度，计算用户行为发生的概率；根据关联度和概率从原始埋点中剔除无效埋点并筛选出新的埋点；将剔除了无效埋点并加入了新的埋点的原始埋点输出。

[0117] 在一个实施例中，上述处理器执行计算机程序时还实现以下步骤：在接收应用程序通过原始埋点收集到的数据之后，还包括：根据预设条件通过数据匹配从数据中剔除不符合预设条件的数据；对剔除了不符合的数据之后获得的数据进行数据分析和数据挖掘；输出数据分析和数据挖掘的结果。

[0118] 在一个实施例中，上述处理器执行计算机程序时还实现以下步骤：在根据数据进行学习，输出学习后获得的埋点之后，还包括：根据业务需求在学习后获得的埋点中添加自定义埋点，构成下一回的原始埋点。

[0119] 在一个实施例中，上述处理器执行计算机程序时还实现以下步骤：一种应用程序埋点方法，包括：接收服务器发送的下一回的原始埋点，下一回的原始埋点包括学习后获得的埋点；将下一回的原始埋点通过动态寻址程序插入到应用程序的类文件中，并打包运行更新后的应用程序。

[0120] 在一个实施例中，上述处理器执行计算机程序时还实现以下步骤：将下一回的原

始埋点通过动态寻址程序插入到应用程序的类文件中，并打包运行更新后的应用程序，包括：将下一回的原始埋点进行结构解析，并调用动态寻址脚本；通过动态寻址脚本对原始埋点解析出的代码结构进行代码循环匹配字节码；通过命令重新打包构建新的应用程序，并运行更新后的应用程序。

[0121] 以上所述实施例仅表达了本发明的几种实施方式，其描述较为具体和详细，但并不能因此而理解为对发明专利范围的限制。应当指出的是，对于本领域的普通技术人员来说，在不脱离本发明构思的前提下，还可以做出若干变形和改进，这些都属于本发明的保护范围。因此，本发明专利的保护范围应以所附权利要求为准。

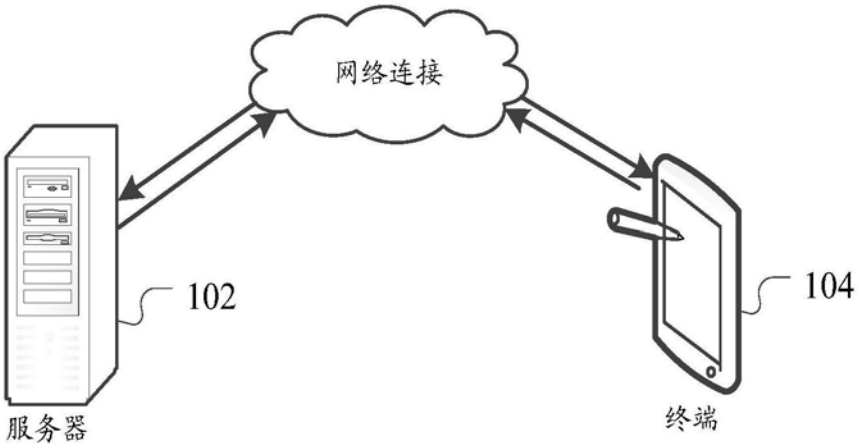

网络连接
102
104
服务器
终端

图1

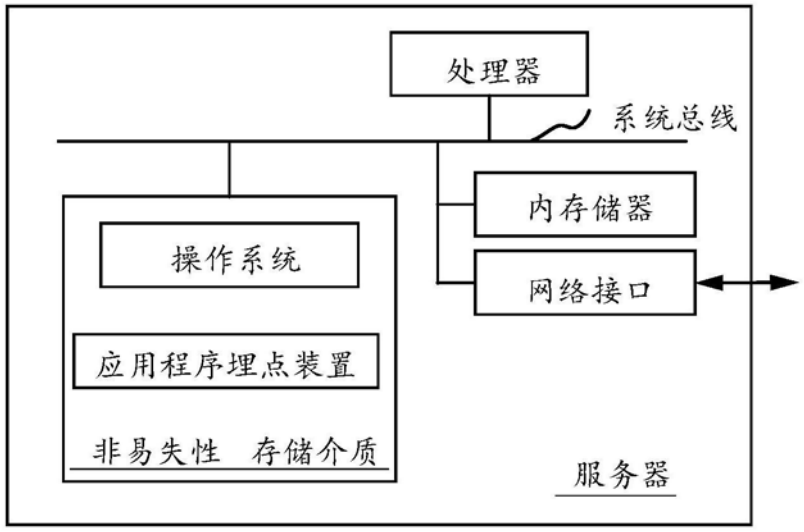

处理器
系统总线
内存储器
网络接口
操作系统
应用程序埋点装置
非易失性 存储介质
服务器

图2

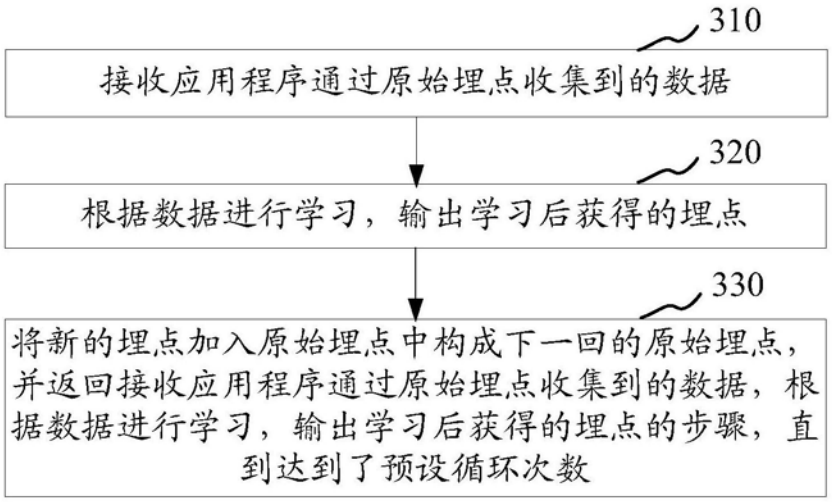

310
接收应用程序通过原始埋点收集到的数据
320
根据数据进行学习，输出学习后获得的埋点
330
将新的埋点加入原始埋点中构成下一回的原始埋点，并返回接收应用程序通过原始埋点收集到的数据，根据数据进行学习，输出学习后获得的埋点的步骤，直到达到了预设循环次数

图3

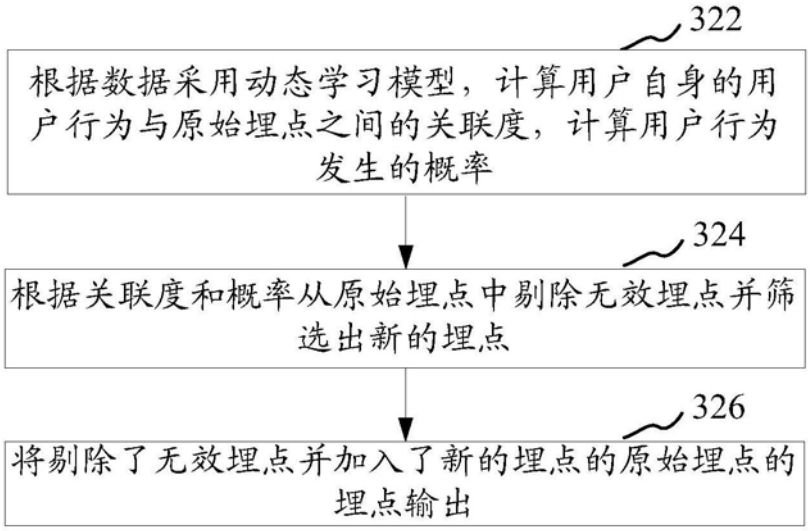

322
根据数据采用动态学习模型，计算用户自身的用户行为与原始埋点之间的关联度，计算用户行为发生的概率
324
根据关联度和概率从原始埋点中剔除无效埋点并筛选出新的埋点
326
将剔除了无效埋点并加入了新的埋点的原始埋点的埋点输出

图4

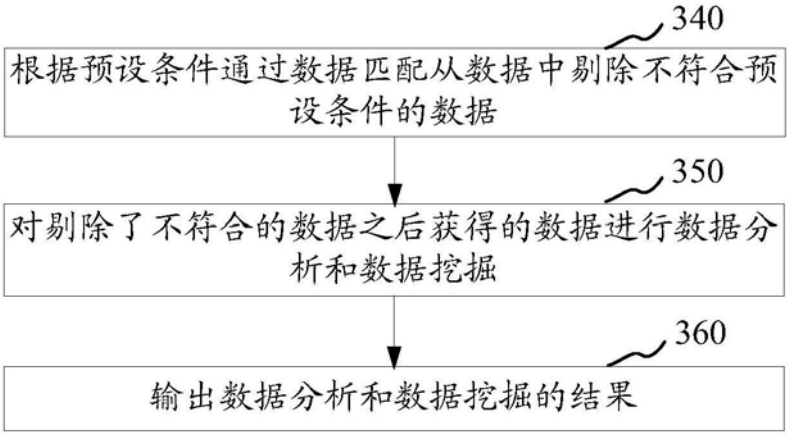

340
根据预设条件通过数据匹配从数据中剔除不符合预设条件的数据
350
对剔除了不符合的数据之后获得的数据进行数据分析和数据挖掘
360
输出数据分析和数据挖掘的结果

图5

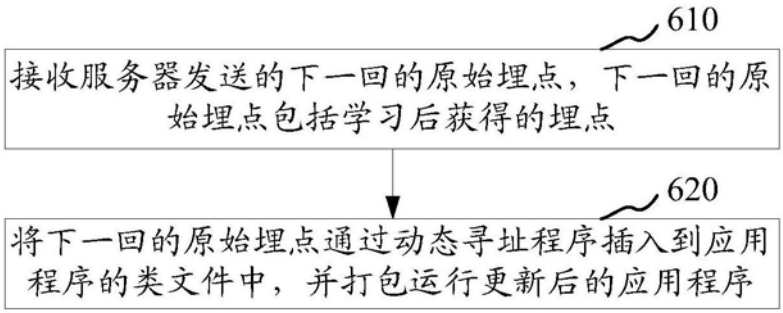

610
接收服务器发送的下一回的原始埋点，下一回的原始埋点包括学习后获得的埋点
620
将下一回的原始埋点通过动态寻址程序插入到应用程序的类文件中，并打包运行更新后的应用程序

图6


应用程序埋点装置700
710
数据接收模块
720
学习模块
730
循环模块

图7

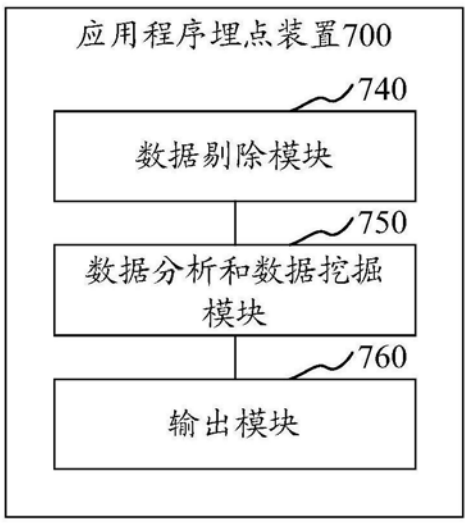

应用程序埋点装置700
740
数据剔除模块
750
数据分析和数据挖掘模块
760
输出模块

图8

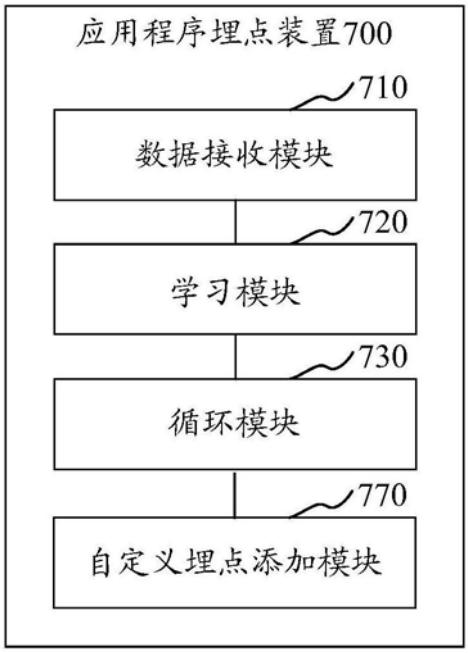

应用程序埋点装置700
710
数据接收模块
720
学习模块
730
循环模块
770
自定义埋点添加模块

图9

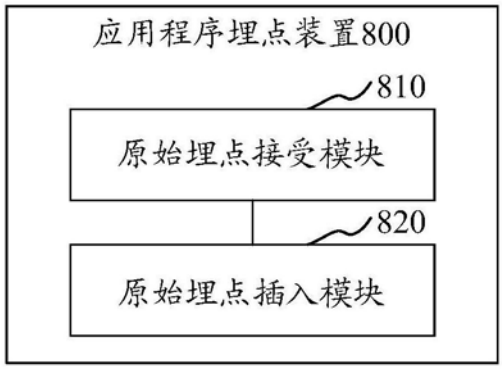


图10